Aufgabe 1

Zeigen Sie für folgendes Programm P

```
x := 5; y := 2; output (x - (y + read))
```

dass sowohl die operationelle Semantik als auch die Reduktionssemantik bei Eingabe $E = (4)$ die Ausgabe $A = (-1)$ bestimmt.
Lösungsidee von Tobi: Wir betrachten die entsprechenden Regeln (aus der Vorlesung) aus beiden Semantiken.
Für die operationelle Semantik:

$$\Delta\langle W|S|I := T.K|E|A\rangle = \langle W|S|T.assign.I.K|E|A\rangle \qquad \text{(OS1a)}$$

$$\Delta\langle n.W|S|assign.I.K|E|A\rangle = \langle W|S[n/I]|K|E|A\rangle, \text{ wobei } n \in ZAHL \text{ und}$$

$$S[n/I](x) = \begin{cases} n, & \text{falls } I = x \\ S(x) & \text{sonst} \end{cases} \qquad \text{(OS1b)}$$

$$\Delta\langle W|S|\underline{read}.K|n.E|A\rangle = \langle n.W|S|K|E|A\rangle \text{ für alle } n \in ZAHL \qquad \text{(OS2)}$$

$$\Delta\langle W|S|\underline{output}\,T.K|E|A\rangle = \langle W|S|T.\underline{output}.K|E|A\rangle \qquad \text{(OS3a)}$$

$$\Delta\langle n.W|S|\underline{output}.K|E|A\rangle = \langle W|S|K|E|n.A\rangle \qquad \text{(OS3b)}$$

$$\Delta\langle W|S|T_1\ \underline{OP}\ T_2.K|E|A\rangle = \langle W|S|T_1.T_2.\underline{OP}.K|E|A\rangle \qquad \text{(OS4a)}$$

$$\Delta\langle n_2.n_1.W|S|\underline{OP}.K|E|A\rangle = \langle(\underline{n_1\ \underline{OP}\ n_2}).W|S|K|E|A\rangle, \text{ falls } \underline{n_1\ \underline{OP}\ n_2}$$
$$\text{nicht aus dem darstellbaren Zahlenbereich herausführt} \qquad \text{(OS4b)}$$

und für die Reduktionssemantik diese Regeln:

$$(x, (s, e, a)) \Rightarrow (s(x), (s, e, a)) \quad \text{, falls } s(x) \neq \underline{\text{frei}} \text{ für } x \in ID, (s, e, a) \in \mathcal{Z} \qquad \text{(RS1a)}$$

$$(I := T, (s, e, a)) \Rightarrow (\underline{skip}, (s[n/I], e', a)) \quad \text{, falls } (T, (s, e, a)) \overset{*}{\Rightarrow} (n, (s, e', a)) \qquad \text{(RS1b)}$$

$$(T_1\ \underline{OP}\ T_2, z) \Rightarrow (n\ \underline{OP}\ T_2, z') \quad \text{, falls } (T_1, z) \overset{*}{\Rightarrow} (n, z') \qquad \text{(RS2a)}$$

$$(n\ \underline{OP}\ T_1, z) \Rightarrow (n\ \underline{OP}\ m, z') \quad \text{, falls } (T, z) \overset{*}{\Rightarrow} (m, z') \qquad \text{(RS2b)}$$

$$(n\ \underline{OP}\ n, z) \Rightarrow (\underline{n\ \underline{OP}\ m}, z') \quad \text{, falls } \underline{n\ \underline{OP}\ m} \in \text{ ZAHL} \qquad \text{(RS2c)}$$

$$\underline{read} \Rightarrow (n, (s, e, a)) \quad \text{, falls } n \in \text{ ZAHL} \qquad \text{(RS3)}$$

$$\underline{output}\ T, (s, e, a) \Rightarrow (\underline{skip}, (s, e', a.n)) \quad \text{, falls } (T, (s, e, a)) \overset{*}{\Rightarrow} (n, (s, e', a)) \qquad \text{(RS4)}$$

Durch auflösen der Klammern erhalten wir folgendes Programm im Kontrollkeller
$K = \{x := 5; y := 2; y + \underline{read}.x. - .\underline{output}\}$,
damit können wir die Kellerspitze simulieren und so die Regeln Schritt für Schritt in folgender Reihenfolge für die operationelle Semantik anwenden:
OS1a, OS1b, OS1a, OS1b, OS4a, OS2, OS4b, OS4a, OS3a, OS3b

Durch induktives anwenden der Regeln für die Reduktionssemantik erhalten das Ergebnis nach anwenden der Regel in dieser Reihenfolge:
RS1b, RS1b, RS3, RS2b, RS2c, RS4

Aufgabe 2

Anmerkung Hinnerk: S hab ich aus der WSKEA Maschine weggelassen, weil es ja eigentlich keinen Speicher und damit kein S gibt. Theoretisch müsste man E und A auch weglassen können, weil es in dieser

Aufgabe ja keine Ein-/Ausgabe gibt. Allerdings wäre ich mir dann spätestens bei der Reduktionssemantik nicht mehr sicher... Für c habe ich auch noch nix tolles...

Gegeben sei folgende Syntax:

```
W := True | False
LOP := AND | OR
LA := W | LA1 LOP LA2 | Not LA
```

zur Formalisierung logischer Ausdrücke.

a) Definieren Sie eine geeignete operationelle Semantik.
$z = \langle W|L.K|E|A\rangle$ mit $L \in LA$

$$\Delta\langle W|\underline{true}.K|E|A\rangle := \langle \underline{true}.W|K|E|A\rangle \tag{OS1}$$

$$\Delta\langle W|\underline{false}.K|E|A\rangle := \langle \underline{false}.W|K|E|A\rangle \tag{OS2}$$

$$\Delta\langle W|\underline{not}\ L.K|E|A\rangle := \langle W|L.\underline{not}\ .K|E|A\rangle \tag{OS3a}$$

$$\Delta\langle l.W|\underline{not}\ .K|E|A\rangle := \langle \neg l.W|K|E|A\rangle, \text{für alle } l \in \{\underline{true}, \underline{false}\}$$

$$\text{wobei } \neg l = \begin{cases} \underline{false} & \text{falls } b = \underline{true} \\ \underline{true} & \text{falls } b = \underline{false} \end{cases} \tag{OS3b}$$

$$\Delta\langle W|LA_1\ \underline{LOP}\ LA_2.K|E|A\rangle := \langle W|LA_1.LA_2.\underline{LOP}.K|E|A\rangle \tag{OS4}$$

$$\Delta\langle l_2.l_1W|\underline{OR}.K|E|A\rangle := \langle \underline{true}.W|K|E|A\rangle, \text{wenn } l_1 = \underline{true} \text{ oder } l_2 = \underline{true} \tag{OS5a}$$

$$\Delta\langle l_2.l_1.W|\underline{OR}.K|E|A\rangle := \langle \underline{false}.W|K|E|A\rangle, \text{wenn } l_1 = \underline{false} \text{ und } l_2 = \underline{false} \tag{OS5b}$$

$$\Delta\langle l_2.l_1.W|\underline{AND}.K|E|A\rangle := \langle \underline{true}.W|K|E|A\rangle, \text{wenn } l_1 = \underline{true} \text{ und } l_2 = \underline{true} \tag{OS6a}$$

$$\Delta\langle l_2.l_1.W|\underline{AND}.K|E|A\rangle := \langle \underline{true}.W|K|E|A\rangle, \text{wenn } l_1 = \underline{false} \text{ und } l_2 = \underline{false} \tag{OS6b}$$

$$\Delta\langle l_2.l_1W|\underline{AND}.K|E|A\rangle := \langle \underline{false}.W|K|E|A\rangle, \text{wenn } l_1 = \underline{false} \text{ oder } l_2 = \underline{false} \tag{OS6c}$$

b) Definieren Sie eine geeignete Reduktionssemantik.
$z = (L, (E, A)$ mit $L \in LA$

$$(\underline{not}\ L, (E, A)) \Rightarrow (\underline{not}\ L', (E', A)), \text{falls } (L, (E, A)) \Rightarrow (L', (E', A)) \tag{RS1a}$$

$$(\underline{not}\ L, (E, A)) \Rightarrow (\neg l, (E, A)), \text{für } l \in \{\underline{true}, \underline{false}\} \tag{RS1b}$$

$$(L_1\ \underline{LOP}\ L_2, (E, A)) \Rightarrow (L_1'\ \underline{LOP}\ L_2, (E', A)), \text{falls } (L_1, (E, A)) \Rightarrow (L_1', (E', A)) \tag{RS2a}$$

$$(l\ \underline{LOP}\ L, (E, A)) \Rightarrow (l\ \underline{LOP}\ L', (E', A)), \text{falls } (L, (E, A)) \Rightarrow (L', (E', A)) \tag{RS2b}$$

$$(l_1\ \underline{AND}\ l_2, (E, A)) \Rightarrow (eval(l_1\ \underline{AND}\ l_2), (E', A)), \text{falls } l_1\ \underline{AND}\ l_2 \text{ ausgewertet werden kann} \tag{RS3}$$

$$(l_1\ \underline{OR}\ l_2, (E, A)) \Rightarrow (eval(l_1\ \underline{OR}\ l_2), (E', A)), \text{falls } l_1\ \underline{OR}\ l_2 \text{ ausgewertet werden kann} \tag{RS4}$$

c) Beweisen Sie die Äquivalenz Ihrer Lösungen zu a) und b).
Lösungsidee von Tobi: Wir definieren einen Ausdruck mit der oben genannten Syntax:

```
True OR Not (False AND False)
```

(Kontrolle: `True`)
... und überprüfen mit Struktureller Induktion die beiden Semantiken, ob wir das gewünschte Ergebnis erhalten.

(a) Durch auflösen der Klammerung erhalten wir folgendes Programm im Kontrollkeller K
$z_0 = \langle W|\texttt{False AND False.Not.True.OR}|E|A\rangle$

$$\begin{aligned}
z_0 &= \langle \epsilon|\texttt{False AND False.Not.True.OR}.\epsilon|\epsilon|\epsilon\rangle \\
&\xRightarrow{\text{OS4}} \langle \epsilon|\texttt{False.False.AND.Not.True.OR}.\epsilon|\epsilon|\epsilon\rangle \\
&\xRightarrow{\text{OS2}} \langle \texttt{False}.\epsilon|\texttt{False.AND.Not.True.OR}.\epsilon|\epsilon|\epsilon\rangle \\
&\xRightarrow{\text{OS2}} \langle \texttt{False.False}.\epsilon|\texttt{AND.Not.True.OR}.\epsilon|\epsilon|\epsilon\rangle \\
&\xRightarrow{\text{OS6b}} \langle \texttt{True}.\epsilon|\texttt{Not.True.OR}.\epsilon|\epsilon|\epsilon\rangle \\
&\xRightarrow{\text{OS3b}} \langle \texttt{False}.\epsilon|\texttt{True.OR}.\epsilon|\epsilon|\epsilon\rangle \\
&\xRightarrow{\text{OS1}} \langle \texttt{True.False}.\epsilon|\texttt{OR}.\epsilon|\epsilon|\epsilon\rangle \\
&\xRightarrow{\text{OS5a}} \langle \texttt{True}.\epsilon|\epsilon|\epsilon|\epsilon\rangle
\end{aligned}$$

(b) some stuff here

Aufgabe 3 (freiwillig)

a) Implementieren Sie die Reduktionssemantik von WHILE in eine Programmiersprache Ihrer Wahl.

b) Implementieren Sie die Semantikfunktion eval, die jeder Programm-Daten-Kombination die entsprechende Ausgabe zuordnet.

c) Testen Sie Ihre Funktion eval am Beispiel des ganzahligen Divisionsprogramms.

Hinweis: Bei Besprechung dieser Aufgabe wird ein Beamer zur Verfügung stehen.